# Wishes Unbox API Documentation

This document provides detailed documentation for the Wishes Unbox API. It includes information about the authentication and profile management endpoints.

## Authentication API Documentation

This section provides an overview of the authentication endpoints for the Wishes Unbox application. It covers sign up, login, forgot password, reset password, Google authentication, and logout.

## Table of Contents

* [Authentication](#authentication)
* [Sign Up](#sign-up)
* [Login](#login)
* [Forgot Password](#forgot-password)
* [Reset Password](#reset-password)
* [Google Auth](#google-auth)
* [Google Callback](#google-callback)
* [Logout](#logout)

## Authentication

### Sign Up

POST /signup

#### Description

Creates a new user account.

#### Request Body Parameters

* firstName (string): The user’s first name.
* lastName (string): The user’s last name.
* email (string): The user’s email address.
* userName (string): The user’s chosen username.
* password (string): The user’s password.
* dateOfBirth (string): The user’s date of birth in ISO 8601 format (YYYY-MM-DD).

#### Example Request

{  
 "firstName": "John",  
 "lastName": "Doe",  
 "email": "john.doe@example.com",  
 "userName": "johndoe",  
 "password": "securepassword",  
 "dateOfBirth": "1990-01-01"  
}

#### Response

* 201 Created: On successful creation of the user account. Returns a JWT token.

### Login

POST /login

#### Description

Authenticates a user and returns a JWT token.

#### Request Body Parameters

* emailOrUsername (string): Either the user’s email address or username.
* password (string): The user’s password.

#### Example Request

{  
 "emailOrUsername": "john.doe@example.com",  
 "password": "securepassword"  
}

#### Response

* 200 OK: On successful authentication. Returns a JWT token.
* 400 Bad Request: If either the email or password is missing.
* 401 Unauthorized: If the provided credentials are invalid.

### Forgot Password

POST /forgotpassword

#### Description

Sends a password reset link to the user’s registered email address.

#### Request Body Parameters

* email (string): The user’s email address.

#### Example Request

{  
 "email": "john.doe@example.com"  
}

#### Response

* 200 OK: If the reset email is successfully sent.
* 404 Not Found: If the user with the given email is not found.

### Reset Password

PUT /resetpassword/:resettoken

#### Description

Resets the user’s password using the provided reset token.

#### Path Parameters

* resettoken (string): The reset token received via email.

#### Request Body Parameters

* password (string): The new password.
* confirmPassword (string): Confirmation of the new password.

#### Example Request

{  
 "password": "newpassword",  
 "confirmPassword": "newpassword"  
}

#### Response

* 200 OK: On successful password reset. Returns a JWT token.
* 400 Bad Request: If the password does not match the confirmation.
* 400 Bad Request: If the reset token is invalid.

### Google Auth

GET /google

#### Description

Redirects the user to the Google OAuth consent screen. Creates or login in user using google’s auth

#### Query Parameters

None.

#### Response

* Redirects the user to the Google OAuth consent screen.

### Google Callback

GET /google/redirect

#### Description

Handles the Google OAuth redirect after consent.

#### Query Parameters

None.

#### Response

* 200 OK: On successful authentication. Returns a JWT token.
* 401 Unauthorized: If the Google authentication fails.

### Logout

GET /logout

#### Description

Logs the user out by clearing the JWT token and sessions.

#### Query Parameters

None.

#### Response

* 200 OK: On successful logout.
* 500 Internal Server Error: If the logout process fails.

## Profile API Documentation

This section provides an overview of the profile management endpoints for the Wishes Unbox application. It covers getting the current user profile, getting a user by ID, updating user profile, getting a user by username, and updating user password.

Please refer to the specific sections for more detailed information on each endpoint.

## Table of Contents

* [Profile Management](#profile-management)
* [Get Current User Profile](#get-current-user-profile)
* [Get User by ID](#get-user-by-id)
* [Update User Profile](#update-user-profile)
* [Get User by Username](#get-user-by-username)
* [Update User Password](#update-user-password)

## Profile Management

### Get Current User Profile

GET /user/me

#### Description

Retrieves the current logged-in user’s profile information.

#### Headers

* Authorization : Cookies

#### Response

* 200 OK: On successful retrieval of the user profile. Returns the user object.

### Get User by ID

GET /user/id/:id

#### Description

Retrieves a user’s profile information by their user ID.

#### Headers

* Authorization : Cookies #### Path Parameters
* id (string): The user ID.

#### Response

* 200 OK: On successful retrieval of the user profile. Returns the user object.
* 404 Not Found: If the user with the given ID is not found.

### Update User Profile

PUT /user/id/:id

#### Description

Updates a user’s profile information.

#### Headers

* Authorization : Cookies

#### Path Parameters

* id (string): The user ID.

#### Request Body Parameters

* firstName (string, optional): The user’s first name.
* lastName (string, optional): The user’s last name.
* email (string, optional): The user’s email address.
* userName (string, optional): The user’s chosen username.
* dateOfBirth (string, optional): The user’s date of birth in ISO 8601 format (YYYY-MM-DD).

#### Response

* 200 OK: On successful update of the user profile. Returns the updated user object.
* 401 Unauthorized: If the user is not authorized to update this profile.
* 404 Not Found: If the user with the given ID is not found.

### Get User by Username

GET /user/:username

#### Description

Retrieves a user’s profile information by their username.

#### Headers

* Authorization : Cookies

#### Path Parameters

* username (string): The user’s username.

#### Response

* 200 OK: On successful retrieval of the user profile. Returns the user object.
* 404 Not Found: If the user with the given username is not found.

### Update User Password

PUT /user/password/update

#### Description

Updates a user’s password.

#### Headers

* Authorization : Cookies

#### Request Body Parameters

* oldPassword (string): The current password.
* newPassword (string): The new password.

#### Response

* 200 OK: On successful password update. Returns a success message.
* 400 Bad Request: If the new password is the same as the old one.
* 401 Unauthorized: If the old password is incorrect.
* 404 Not Found: If the user is not found.